

LAB 01

COMPUTER ORGANIZATION AND ASSEMBLY LANG(COAL)



Syed Muhammad Faheem

STUDENT NAME

20K-1054

ROLL NO

3E

SEC

SIGNATURE & DATE

MARKS AWARDED: _____

NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES
(NUCES), KARACHI

Prepared by: Amin Sadiq

Version: 1.0

Date:

Lab Session 01 CONFIGURATION OF VS 2019

Lab Session 01: CONFIGURATION OF VS 2019

Objectives:

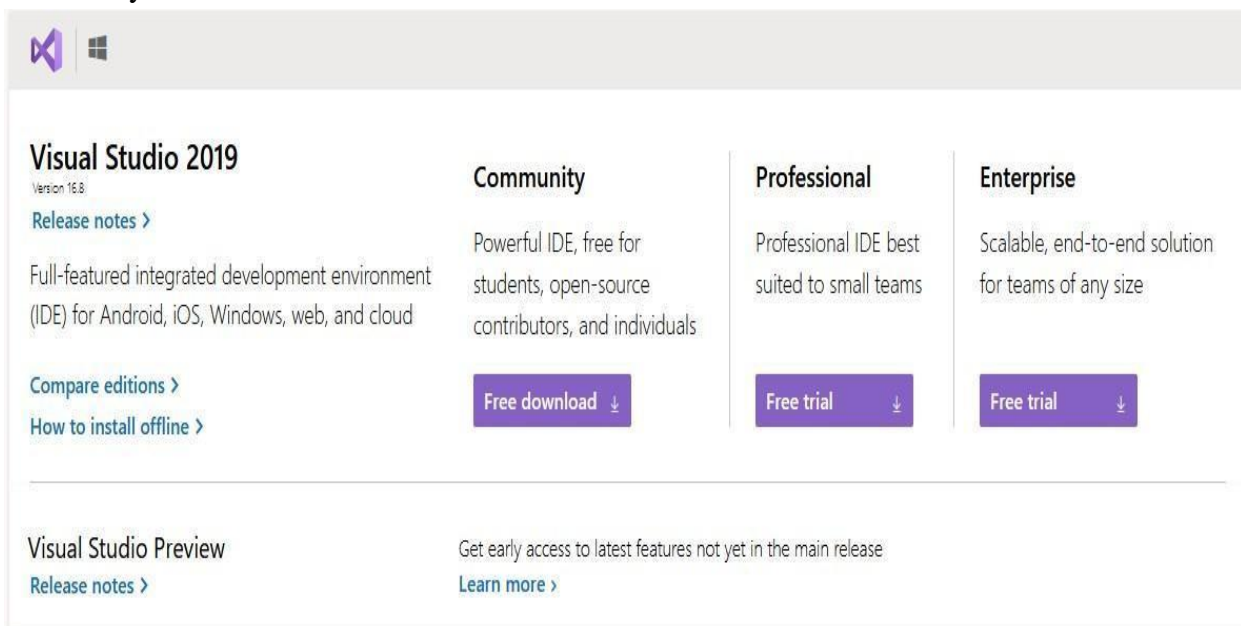
- Introduction to Visual Studio 2019
- An introduction to Assembly Language
- Understanding the Visual Studio 2019
- Configuring Visual Studio 2019 to activate MASM assembler
- Running a test program

SECTION 1: INTRODUCTION TO VISUAL STUDIO 2019

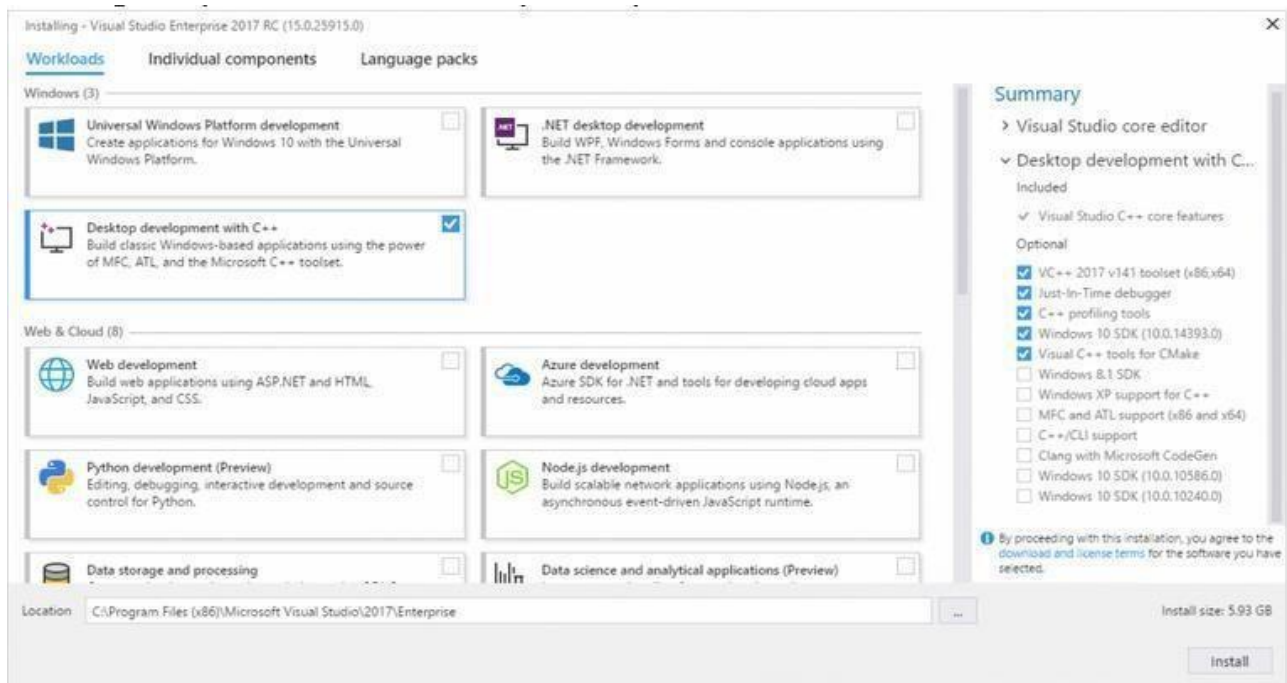
Visual Studio (for all its versions) is an integrated development environment (IDE) from Microsoft. It is a leading tool to develop computer programs, as well as web sites, web applications and web services. For this course, we will use Visual Studio version 2019 to develop programs in Assembly Language. We could, however, use a stand-alone assembler like NASM or MASM to code in Assembly Language.

INSTALLATION PROCESS

Go to this link <https://visualstudio.microsoft.com/downloads/> and select VS 2019 Download for community version



Run that downloaded setup on your system and when it's complete, you have to download and install **Desktop Development with C++**. When it's done you are ready to go.

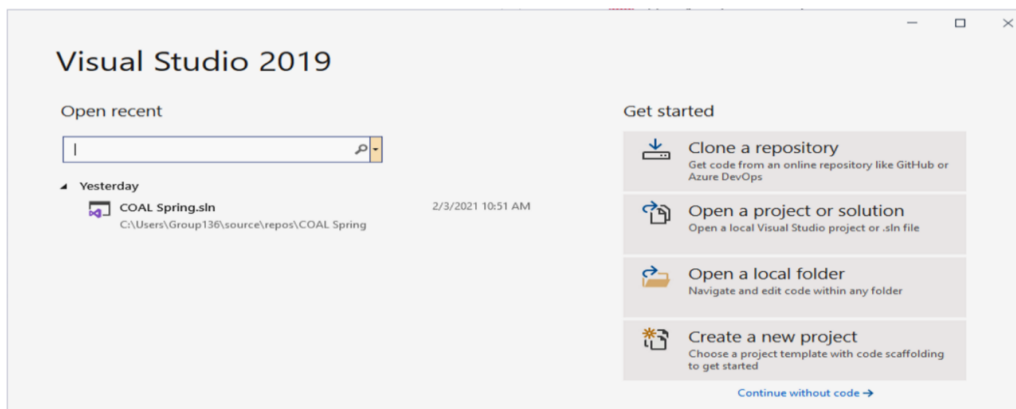


CONFIGURATION VS2019 FOR ASSEMBLY LANGUAGE

Click [here](http://www.asmirvine.com/gettingStartedVS2019/Irvine.zip) to download Irvine library from this link: www.asmirvine.com/gettingStartedVS2019/Irvine.zip

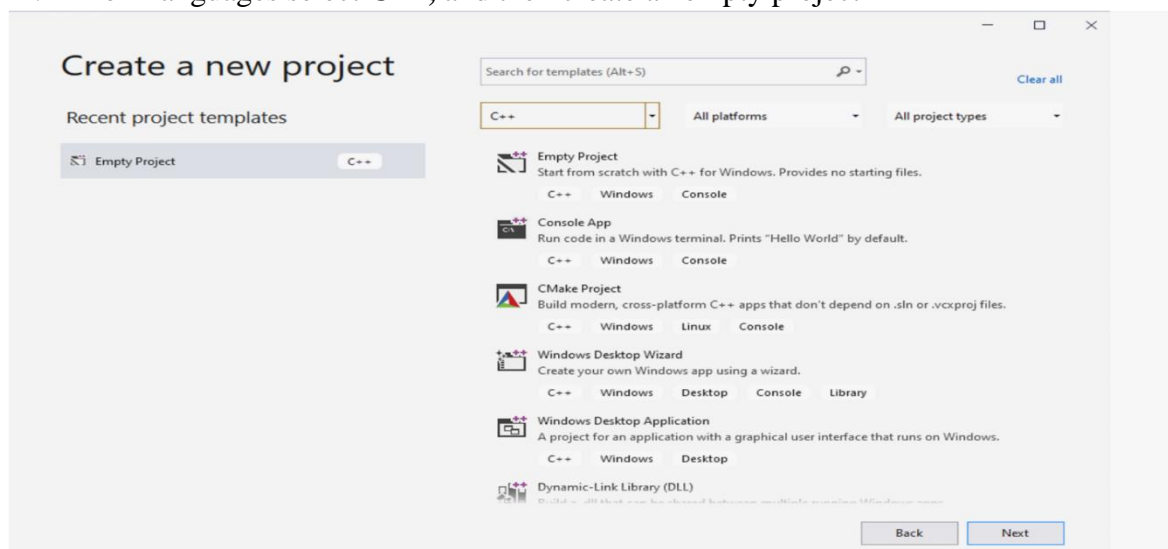
Once you have downloaded the required Irvine library, install it in your computer and verify that a folder named Irvine has been created in your C:\ drive. Now, follow these steps to configure Visual Studio 2019:

1. Start Microsoft Visual Studio 2019. If you are running it for the first time then this would be the screen you may see



Select
Create a
new project.

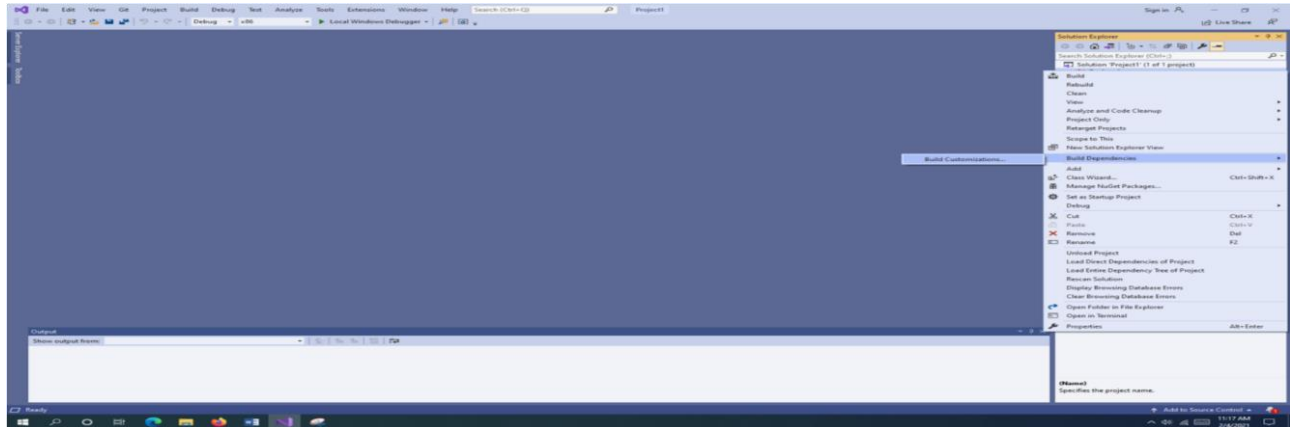
2. From languages select C++, and then create an empty project



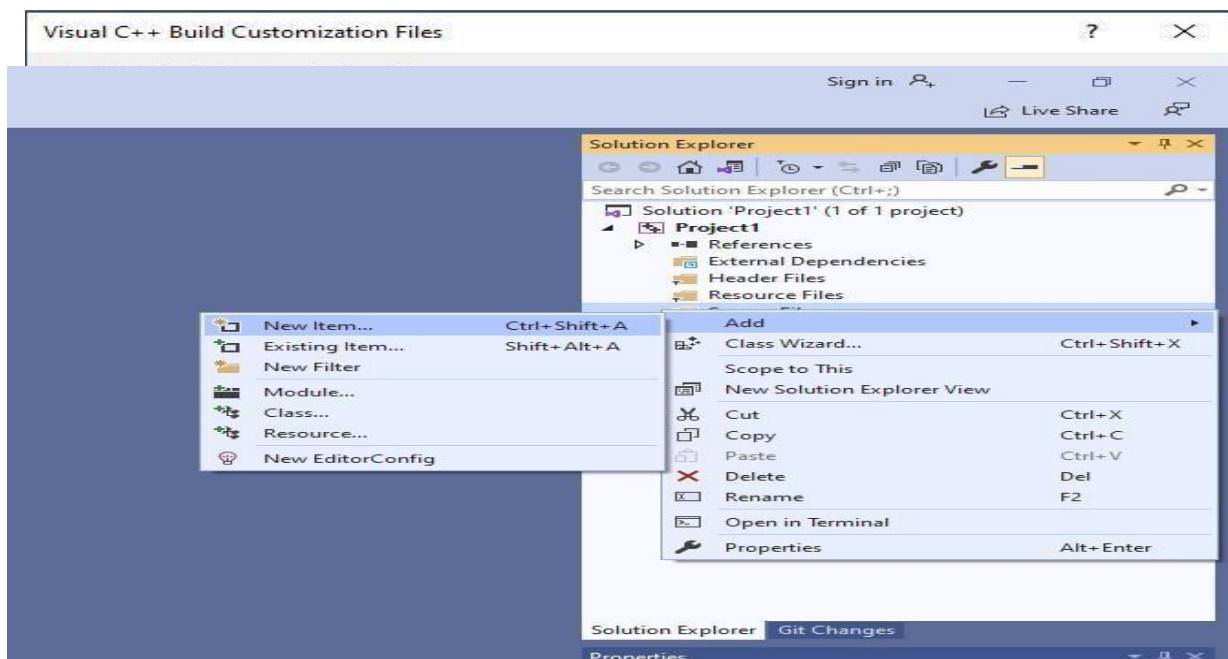
3. Once your new project is created, press **Ctrl+Alt+L** to open **Solution Explorer**. In the solution explorer window, you would see your project's file hierarchy.



Now right click on your project. Go to **Build Dependencies** and then select **Build Customization**

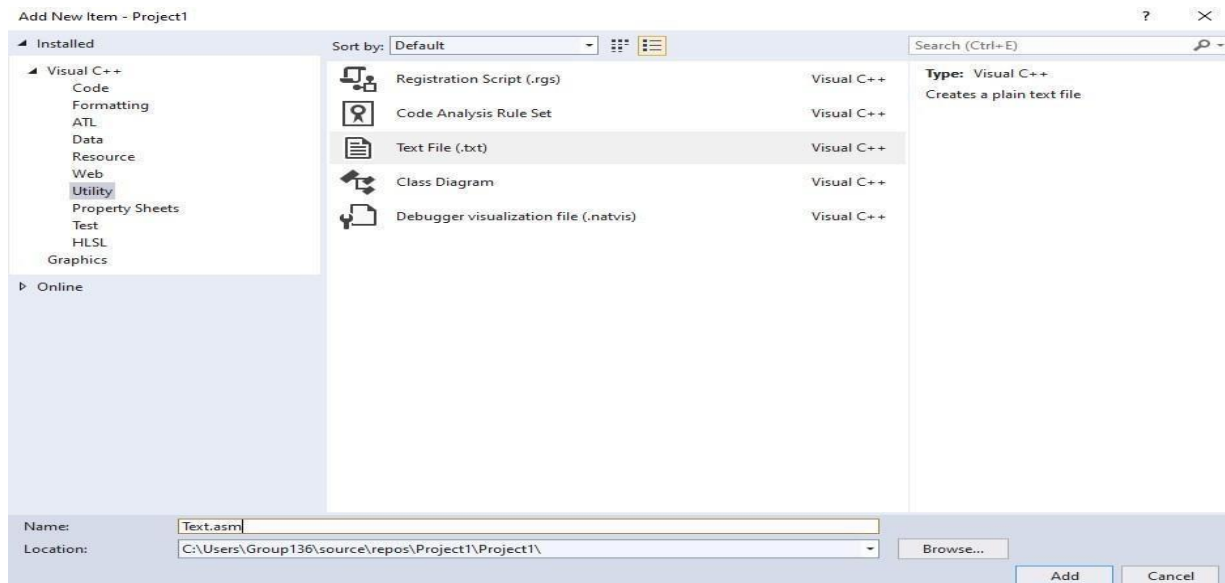


Tick the **masm** checkbox & select **OK**.

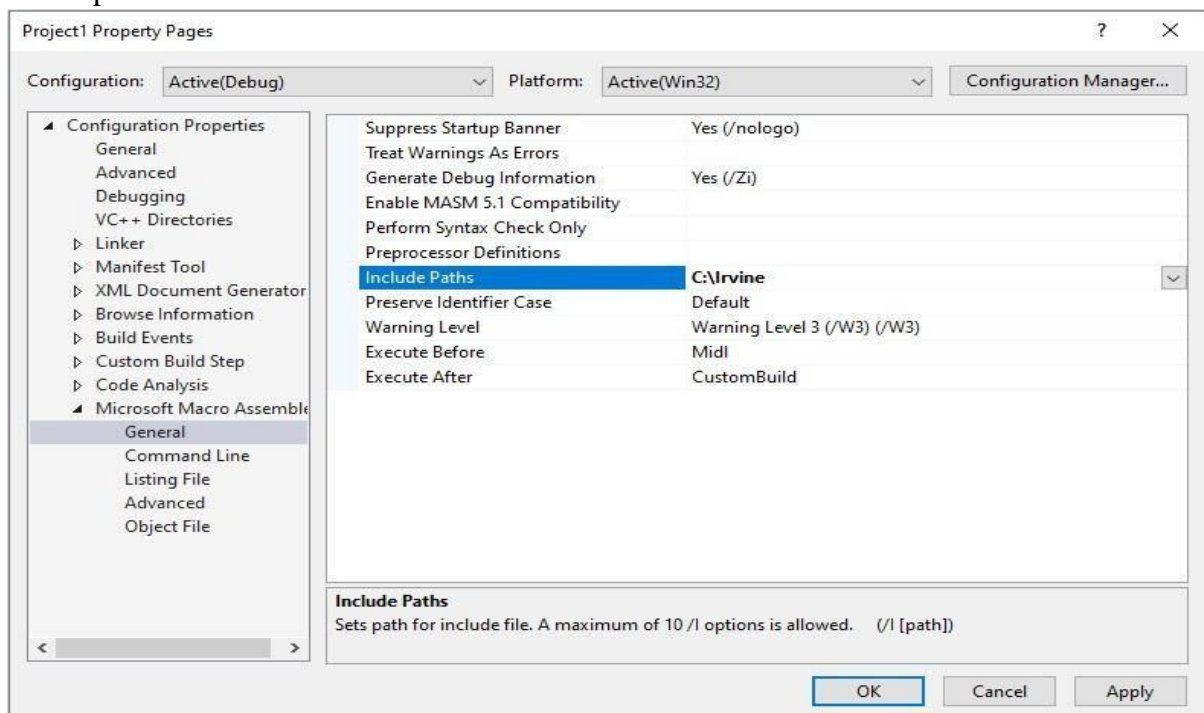


4. Right-click on **Source Files** in solution explorer & select **Add > New Item**.

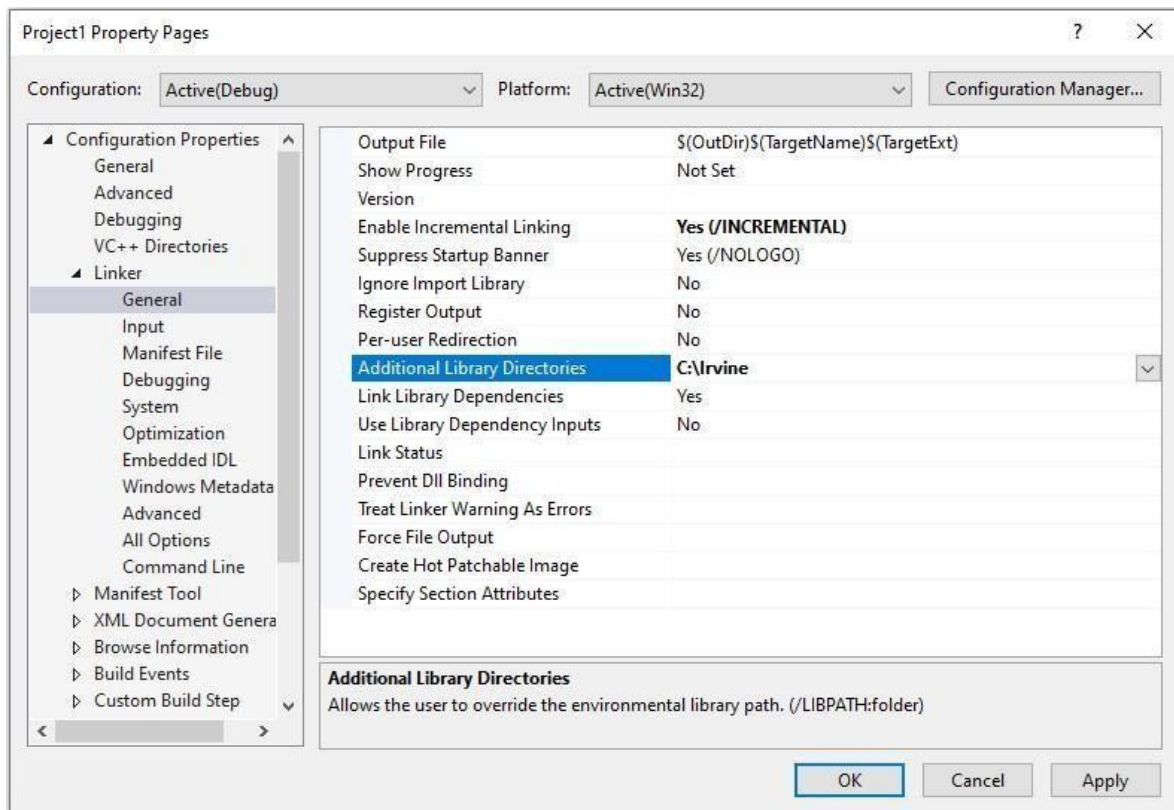
Now go to **Utility > Text File** to add a new file, but we do not want to add .txt file, instead we want to add a .asm file. So, rename your new text file as Test.asm (we can choose any other name e.g. xyz.asm but for this tutorial we will use the name Test.asm).



- Now right-click your project again and click **Properties**. Now click the tiny arrow marker besides **Configuration Properties** to expand it. Now click **Microsoft Macro Assembler** and expand it.



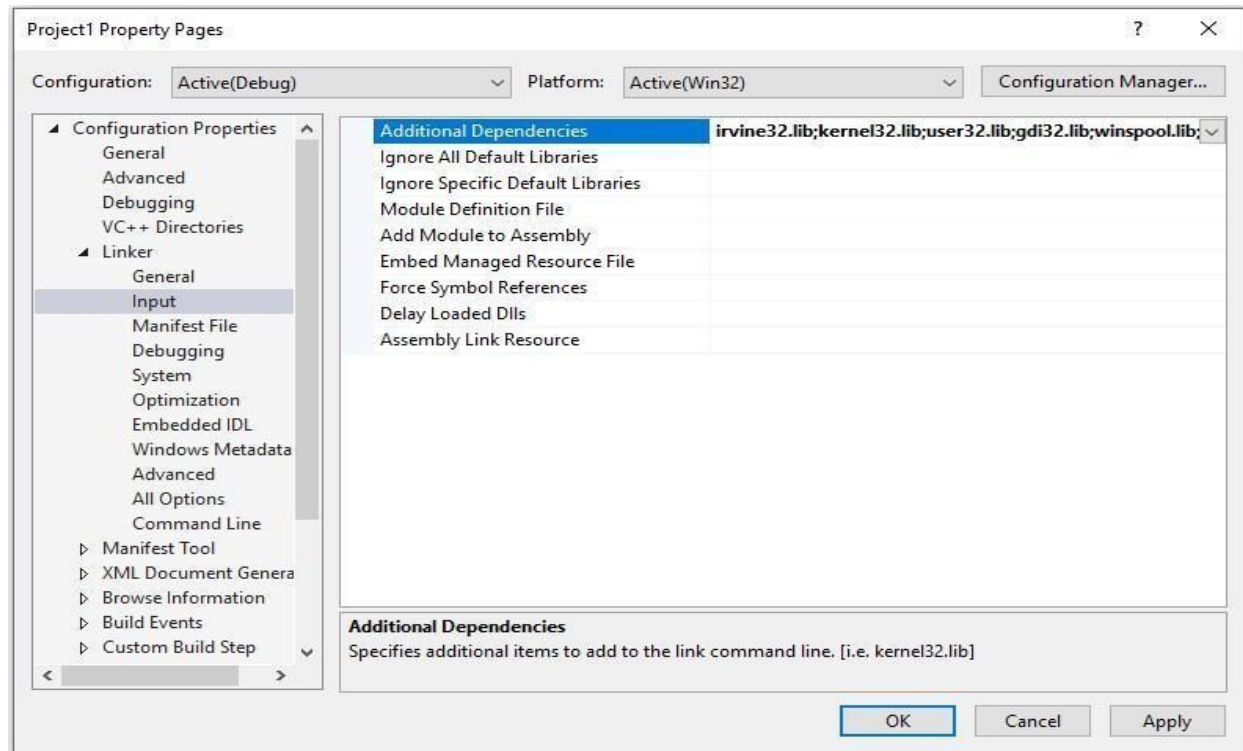
- Now click **General** entry under Microsoft Macro Assembler and then set the value of **Include Paths** as **C:\Irvine**. The menu should now like this.
- Click **Linker** tab to expand it. Select **General** and set the value of **Additional Library Directories** to **C:\Irvine**



Our Visual Studio 2019 configuration for Assembly Language is complete. We can now write a sample program and run it to test our project. Open Test.asm from the solution explorer by double-clicking it. The Test.asm file will contain all the code that we write in our program. Go on and copy the following code onto your Test.asm file.

```
TITLE My First Program (Test.asm)
INCLUDE Irvine32.inc
```

8. Click **Input**, select **Additional Dependencies**. You will see a list of different .lib file names written there, do not alter any of those. Write **irvine32.lib**; at the start of the list like this.



```

Microsoft Visual Studio Debug Console

EAX=00000010  EBX=00000025  ECX=005910AA  EDX=005910AA
ESI=005910AA  EDI=005910AA  EBP=00FDFAEC  ESP=00FDFAE0
EIP=0059366F  EFL=00000246  CF=0  SF=0  ZF=1  OF=0  AF=0  PF=1

C:\Users\Group136\source\repos\Project1\Debug\Project1.exe (process 10308) exited with code 0.
Press any key to close this window . . .

```

.code main


```
PROC mov
    eax, 10h mov
    ebx, 25h call
DumpRegs
exit
main ENDP
END main
```

Press **Ctrl+F5** to see the output in console window.

As we can see in the output window, the program has affected two registers `eax` & `ebx`. Let us dissect our code line by line to see what it does.

The first line `TITLE MyFirstProgram (Test.asm)` gives an optional title to our program. The second line `INCLUDE irvine32.inc` adds a reference to the include file that links your program to the Irvine library. The third line `.code` defines the beginning of the code segment (to be covered in detail later). The code segment is the segment of memory where all your code resides. In the fourth line, a main procedure is defined. The fifth and sixth lines show a mnemonic `mov` (to be covered in detail later) that ‘moves’ values `10h` and `25h` to `eax` and `ebx`, respectively. The radix `h` defines a hexadecimal constant.

The lines seven and eight calls the procedure `DumpRegs` that outputs the current values of the registers followed by a call to windows procedure named `exit` that halts the program. The lines nine and ten mark the end of the main procedure.

SECTION 2: DEBUGGING OUR PROGRAM

We have seen how to configure Visual Studio 2019 for Assembly Language and tested it with a sample program. The output of our sample program was displayed using a console window but it is usually more desirable to watch the step by step execution of our program with each line of code using breakpoints.

Let us briefly define the keywords relevant to debugging in Visual Studio and then we will cover an example for understanding.

DEBUGGER

The (Visual Studio) debugger helps us observe the run-time behavior of our program and find problems. With the debugger, we can break execution of our program to examine our code, examine and edit variables, view registers, see the instructions created from our source code, and view the memory space used by our application.



BREAKPOINT

A breakpoint is a signal that tells the debugger to temporarily suspend execution of your program at a certain point. When execution is suspended at a breakpoint, your program is said to be in break mode.

CODE STEPPING

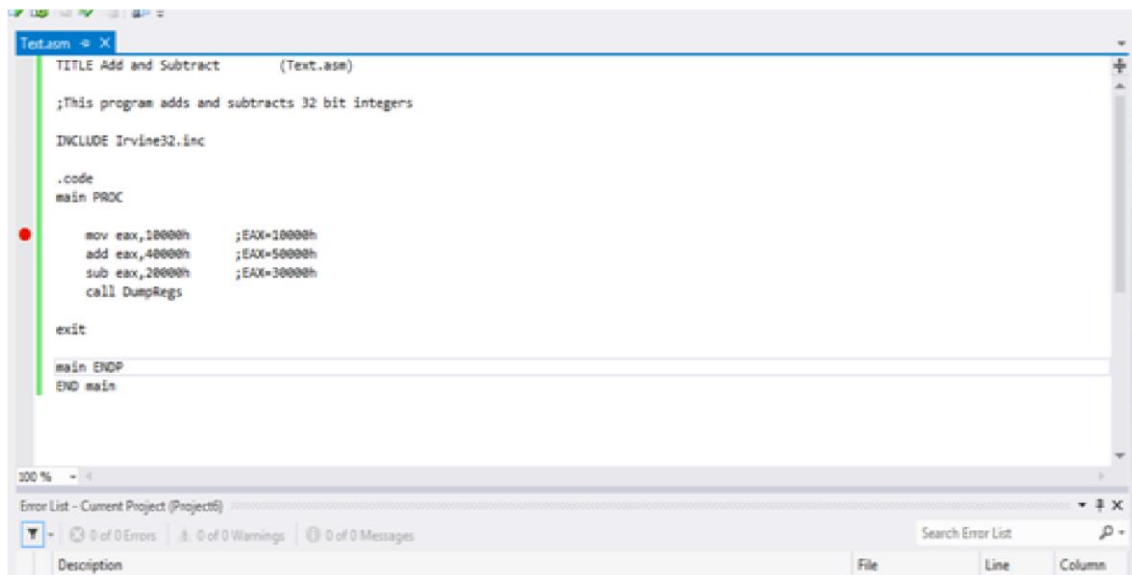
One of the most common debugging procedures is stepping: executing code one line at a time. The Debug menu provides three commands for stepping through code:

- Step Into (By pressing F11)
- Step Over (By pressing F10)
- Step Out (Shift+F11)

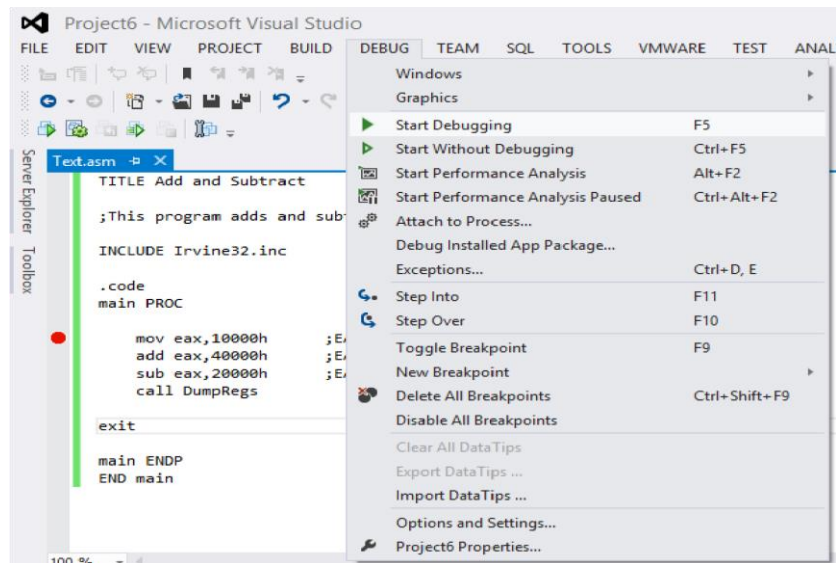
SINGLE STEPPING

To see the values of internal registers and memory variables during execution, let us use an example. Copy the following code onto your Test.asm file.k

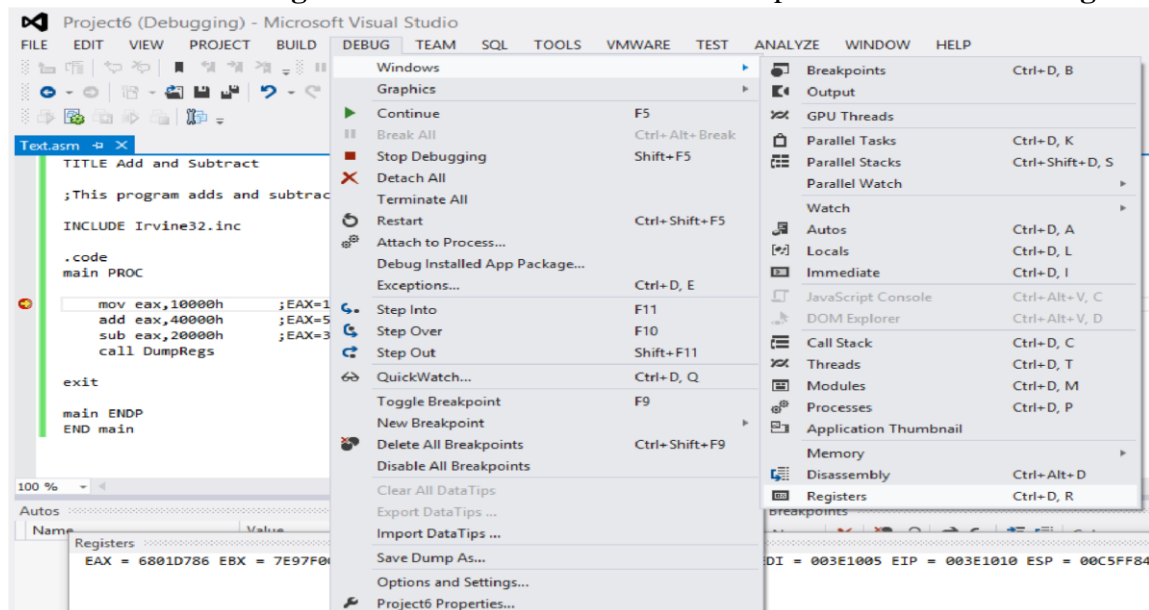
1. Right-click on line 6 to insert a breakpoint.



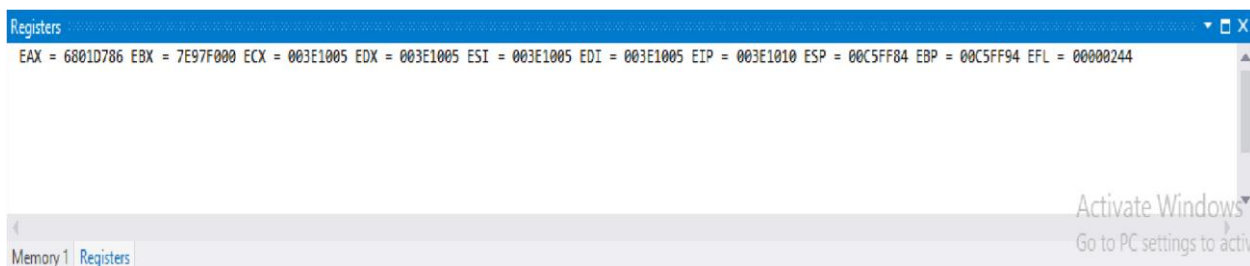
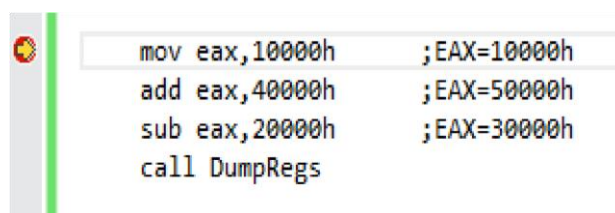
2. Click on **Debug** tab from the toolbar, select **Start Debugging** OR press **F10** to start stepping over the code.



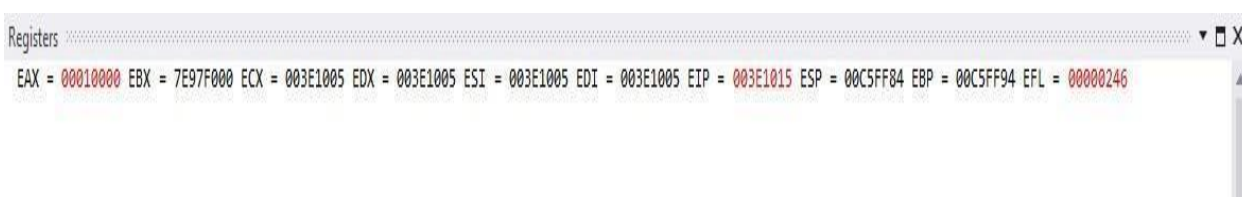
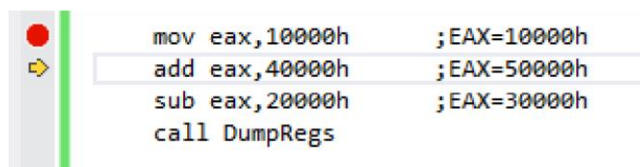
3. Click on **Debug** tab than select Windows after that open menu and select **Registers** option.



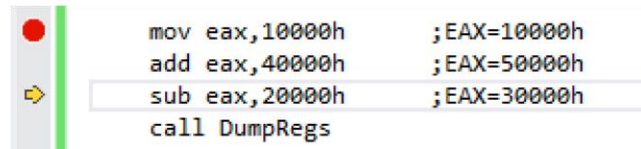
4. Breakpoint set on 1st instruction



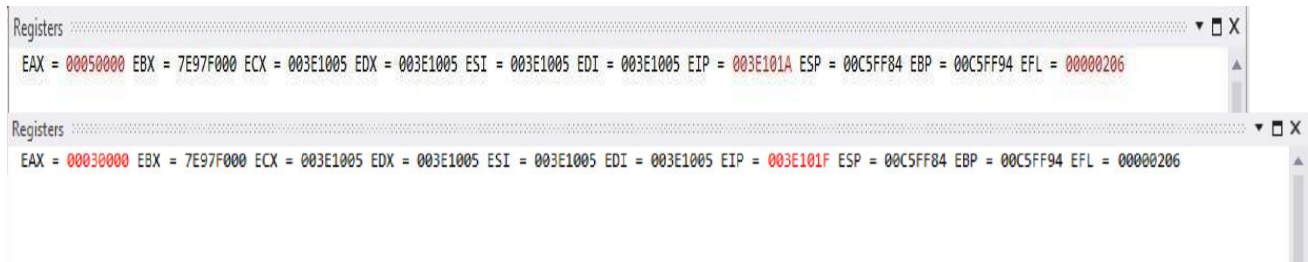
Press **F10** again to execute next line.



Again press **F10** key for next instruction execution.



```
mov eax,10000h    ; EAX=10000h
add eax,40000h    ; EAX=50000h
sub eax,20000h    ; EAX=30000h
call DumpRegs
```



Registers

EAX = 00050000 EBX = 7E97F000 ECX = 003E1005 EDX = 003E1005 ESI = 003E1005 EDI = 003E1005 EIP = 003E101A ESP = 00C5FF84 EBP = 00C5FF94 EFL = 00000206

Registers

EAX = 00030000 EBX = 7E97F000 ECX = 003E1005 EDX = 003E1005 ESI = 003E1005 EDI = 003E1005 EIP = 003E101F ESP = 00C5FF84 EBP = 00C5FF94 EFL = 00000206

Press **F10** again, the program will not terminate after executing the current instruction and as soon as it reaches the line with a call to **DumpRegs**

SECTION 2: EXERCISE

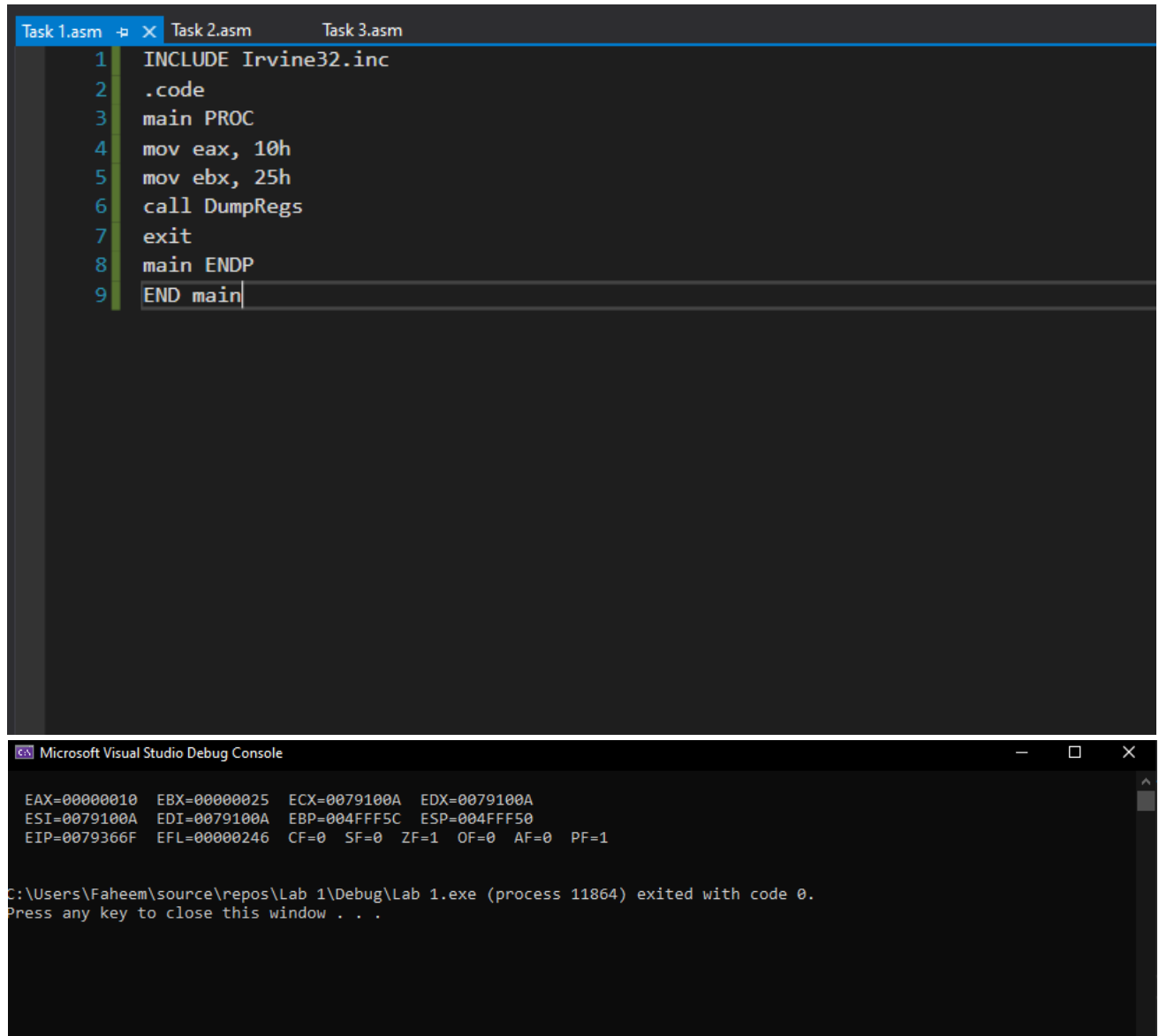
1. Install Visual Studio 2019 & create a new Visual C++ project for Assembly Language.
2. Configure the project using the steps show in this lab.
3. Run a test program in console window by changing the value of EAX in line 6 to 8500h.
4. Debug the below program and note down the values of all the registers after the execution of each line.

```
TITLE My First Program (Test.asm)
INCLUDE Irvine32.inc
```

```
.code
main PROC
    mov eax, 47h
    mov ebx, 39h
    mov ecx, 60h
    add eax, ebx
    add eax, ecx
    mov ebx, 85h
    mov ecx, 64h
    add eax, ebx
    add eax, ecx
```

```
call DumpRegs
exit
main ENDP
END main
```

Task 1:



The screenshot displays the Visual Studio IDE with three tabs: Task 1.asm, Task 2.asm, and Task 3.asm. The active tab, Task 1.asm, contains the following assembly code:

```
1  INCLUDE Irvine32.inc
2  .code
3  main PROC
4  mov eax, 10h
5  mov ebx, 25h
6  call DumpRegs
7  exit
8  main ENDP
9  END main
```

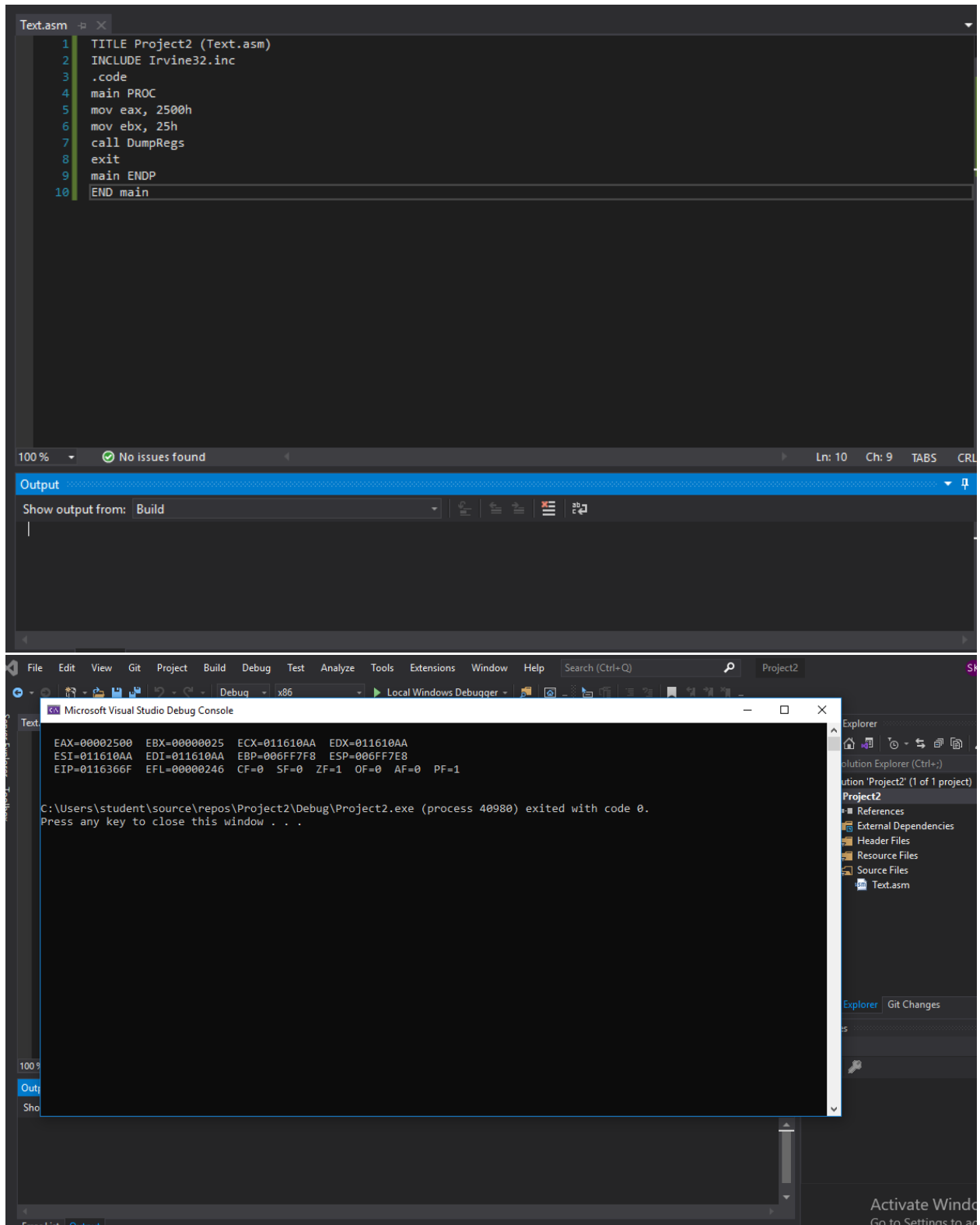
Below the code editor, the Microsoft Visual Studio Debug Console shows the execution output:

```
EAX=00000010  EBX=00000025  ECX=0079100A  EDX=0079100A
ESI=0079100A  EDI=0079100A  EBP=004FFF5C  ESP=004FFF50
EIP=0079366F  EFL=00000246  CF=0  SF=0  ZF=1  OF=0  AF=0  PF=1

C:\Users\Faheem\source\repos\Lab 1\Debug\Lab 1.exe (process 11864) exited with code 0.
Press any key to close this window . . .
```


Task 2:

Task 2 includes Run a test program in console window by changing the value of EAX in line 6 to 8500h.



Task 3:

```
Task 3.asm  -p X
1  INCLUDE Irvine32.inc
2  .code
3  main PROC
4  mov eax, 47h
5  mov ebx, 39h
6  mov ecx, 60h
7  add eax, ebx
8  add eax, ecx
9  mov ebx, 85h
10 mov ecx, 64h
11 add eax, ebx
12 add eax, ecx
13 call DumpRegs
14 exit
15 main ENDP
16 END main
```

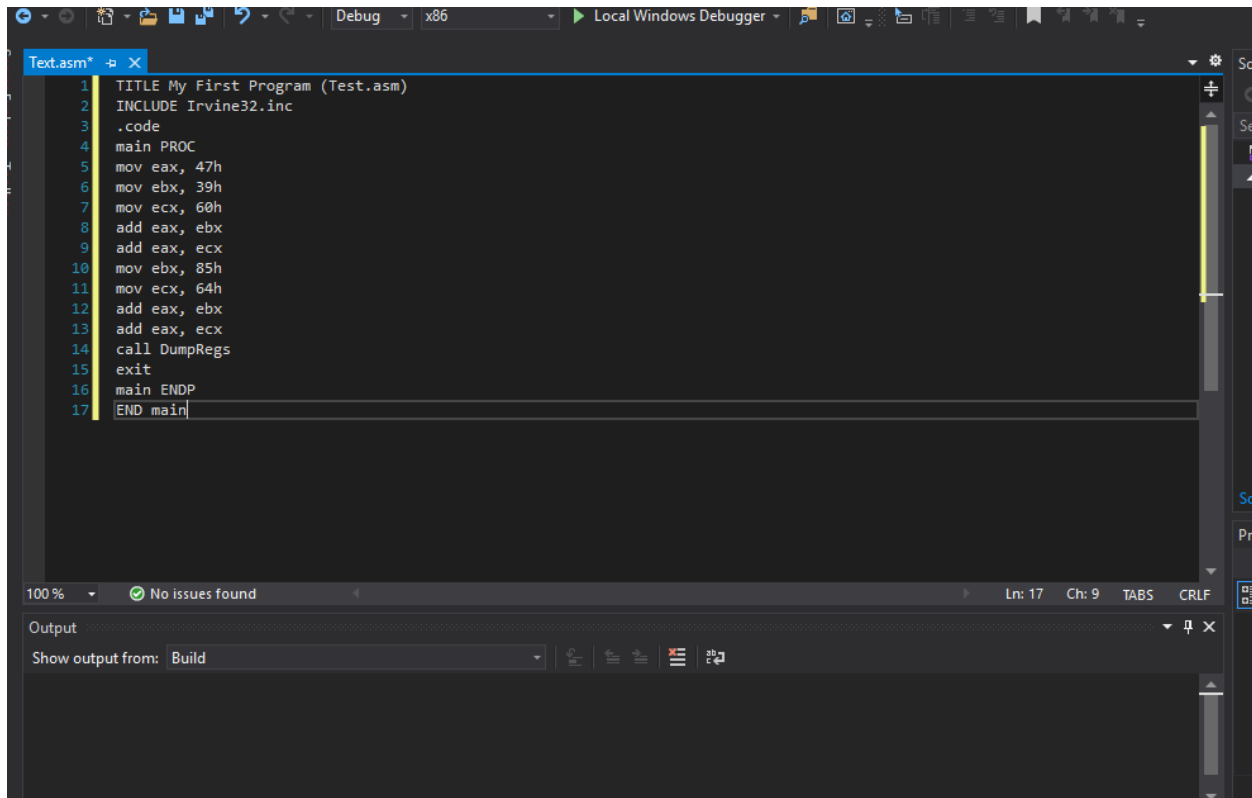
```
Microsoft Visual Studio Debug Console

EAX=000001C9  EBX=00000085  ECX=00000064  EDX=002E100A
ESI=002E100A  EDI=002E100A  EBP=006FFE14  ESP=006FFE08
EIP=002E3686  EFL=00000206  CF=0  SF=0  ZF=0  OF=0  AF=0  PF=1

C:\Users\Faheem\source\repos\Lab 1\Debug\Lab 1.exe (process 9972) exited with code 0.
Press any key to close this window . . .
```

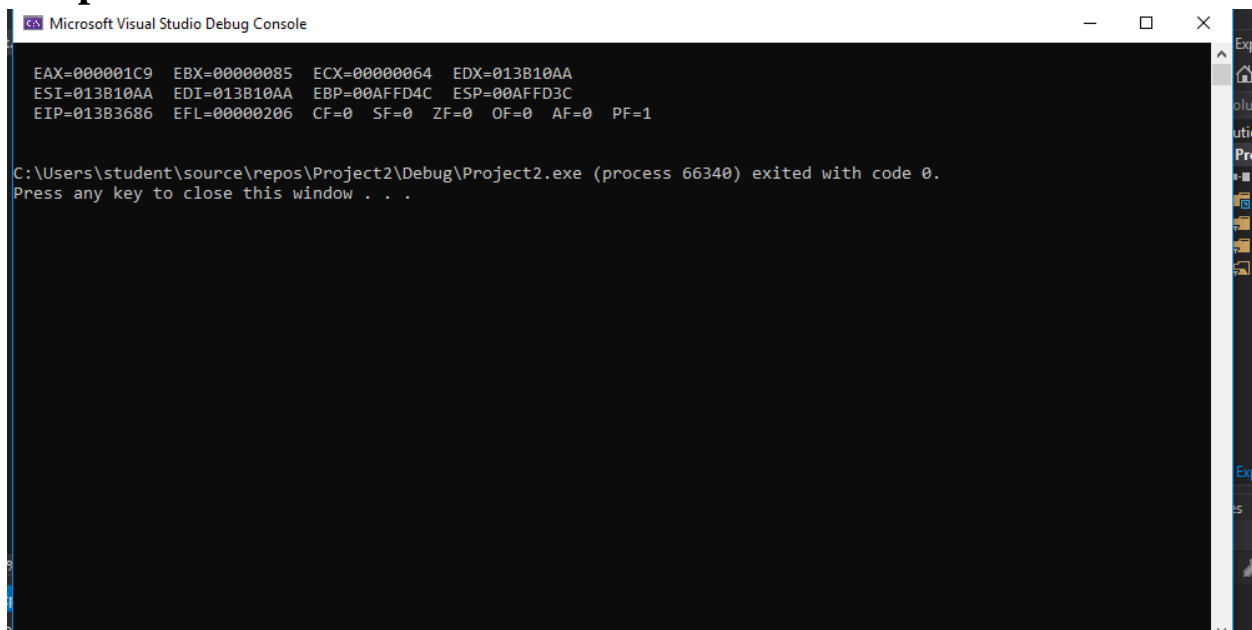
DEBUGGING TASK

Code:



```
1 TITLE My First Program (Test.asm)
2 INCLUDE Irvine32.inc
3 .code
4 main PROC
5 mov eax, 47h
6 mov ebx, 39h
7 mov ecx, 60h
8 add eax, ebx
9 add eax, ecx
10 mov ebx, 85h
11 mov ecx, 64h
12 add eax, ebx
13 add eax, ecx
14 call DumpRegs
15 exit
16 main ENDP
17 END main
```

Output:



```
EAX=000001C9  EBX=00000085  ECX=00000064  EDX=013B10AA
ESI=013B10AA  EDI=013B10AA  EBP=00AFFD4C  ESP=00AFFD3C
EIP=013B3686  EFL=00000206  CF=0   SF=0   ZF=0   OF=0   AF=0   PF=1

C:\Users\student\source\repos\Project2\Debug\Project2.exe (process 66340) exited with code 0.
Press any key to close this window . . .
```

Debugging now:

Registers
EAX = 00000047 EBX = 010A0000 ECX = 013B10AA EDX = 013B10AA ESI = 013B10AA EDI = 013B10AA EIP = 01383665 ESP = 0135F814 EBP = 0135F824 EFL = 00000246

100%
Text.asm
1 TITLE My First Program (Test.asm)
2 INCLUDE Irvine32.inc
3 .code
4 main PROC
5 mov eax, 47h
6 mov ebx, 39h
7 mov ecx, 60h
8 add eax, ebx
9 add eax, ecx
10 mov ebx, 85h
11 mov ecx, 64h
12 add eax, ebx
13 add eax, ecx
14 call DumpRegs

Diagnostic Tools
Diagnostics session: 0 seconds (13 m
8ms
Events
Process Memory
Summary Events Memory Usage
Events
Show Events (2 of 3)
Memory Usage

Registers
EAX = 00000047 EBX = 00000039 ECX = 013B10AA EDX = 013B10AA ESI = 013B10AA EDI = 013B10AA EIP = 0138366A ESP = 0135F814 EBP = 0135F824 EFL = 00000246

100%
Text.asm
1 TITLE My First Program (Test.asm)
2 INCLUDE Irvine32.inc
3 .code
4 main PROC
5 mov eax, 47h
6 mov ebx, 39h
7 mov ecx, 60h
8 add eax, ebx
9 add eax, ecx
10 mov ebx, 85h
11 mov ecx, 64h
12 add eax, ebx
13 add eax, ecx
14 call DumpRegs

Diagnostic Tools
Diagnostics session: 0 seconds (13 m
8ms
Events
Process Memory
Summary Events Memory Usage
Events
Show Events (4 of 4)
Memory Usage

Registers
EAX = 00000047 EBX = 00000039 ECX = 00000060 EDX = 013B10AA ESI = 013B10AA EDI = 013B10AA EIP = 0138366F ESP = 0135F814 EBP = 0135F824 EFL = 00000246

100%
Text.asm
1 TITLE My First Program (Test.asm)
2 INCLUDE Irvine32.inc
3 .code
4 main PROC
5 mov eax, 47h
6 mov ebx, 39h
7 mov ecx, 60h
8 add eax, ebx
9 add eax, ecx
10 mov ebx, 85h
11 mov ecx, 64h
12 add eax, ebx
13 add eax, ecx
14 call DumpRegs

Diagnostic Tools
Diagnostics session: 0 seconds (13 m
8ms
Events
Process Memory
Summary Events Memory Usage
Events
Show Events (4 of 5)
Memory Usage

Output

The image displays three sequential screenshots of a debugger window, showing the execution of an assembly program. The assembly code is as follows:

```
1 TITLE My First Program (Test.asm)
2 INCLUDE Irvine32.inc
3 .code
4 main PROC
5 mov eax, 47h
6 mov ebx, 39h
7 mov ecx, 60h
8 add eax, ebx
9 add eax, ecx
10 mov ebx, 85h
11 mov ecx, 64h
12 add eax, ebx
13 add eax, ecx
14 call DumpRegs
```

Screenshot 1 (Top): The debugger shows the initial state. The registers window displays: EAX = 00000000, EBX = 00000039, ECX = 00000060, EDX = 013B10AA, ESI = 013B10AA, EDI = 013B10AA, EIP = 013B3671, ESP = 0135F814, EBP = 0135F824, EFL = 00000212. The instruction pointer (EIP) is highlighted at 013B3671.

Screenshot 2 (Middle): The debugger shows execution at line 10. The registers window displays: EAX = 000000E0, EBX = 00000039, ECX = 00000060, EDX = 013B10AA, ESI = 013B10AA, EDI = 013B10AA, EIP = 013B3673, ESP = 0135F814, EBP = 0135F824, EFL = 00000202. The instruction pointer (EIP) is highlighted at 013B3673.

Screenshot 3 (Bottom): The debugger shows execution at line 11. The registers window displays: EAX = 000000E0, EBX = 00000085, ECX = 00000060, EDX = 013B10AA, ESI = 013B10AA, EDI = 013B10AA, EIP = 013B3678, ESP = 0135F814, EBP = 0135F824, EFL = 00000202. The instruction pointer (EIP) is highlighted at 013B3678.

The image displays two screenshots of the Visual Studio IDE, showing the execution of an assembly program. The top screenshot shows the program at line 12, and the bottom screenshot shows it at line 13. Both screenshots show the assembly code, registers, and diagnostic tools.

Top Screenshot:

- Registers:** EAX = 000000E0, EBX = 00000085, ECX = 00000064, EDX = 013B10AA, ESI = 013B10AA, EDI = 013B10AA, EIP = 013B367D, ESP = 0135F814, EBP = 0135F824, EFL = 00000202.
- Assembly Code:**

```
1 TITLE My First Program (Test.asm)
2 INCLUDE Irvine32.inc
3 .code
4 main PROC
5 mov eax, 47h
6 mov ebx, 39h
7 mov ecx, 60h
8 add eax, ebx
9 add eax, ecx
10 mov ebx, 85h
11 mov ecx, 64h
12 add eax, ebx 51ms elapsed
13 add eax, ecx
14 call DumpRegs
```
- Diagnostic Tools:** Diagnostics session: 0 seconds (14 ms selected). Events: 11ms, 12ms. Process Memory: Summary, Events, Memory Usage, CPU Usage. Show Events (9 of 9).

Bottom Screenshot:

- Registers:** EAX = 00000165, EBX = 00000085, ECX = 00000064, EDX = 013B10AA, ESI = 013B10AA, EDI = 013B10AA, EIP = 013B367F, ESP = 0135F814, EBP = 0135F824, EFL = 00000206.
- Assembly Code:**

```
1 TITLE My First Program (Test.asm)
2 INCLUDE Irvine32.inc
3 .code
4 main PROC
5 mov eax, 47h
6 mov ebx, 39h
7 mov ecx, 60h
8 add eax, ebx
9 add eax, ecx
10 mov ebx, 85h
11 mov ecx, 64h
12 add eax, ebx
13 add eax, ecx 51ms elapsed
14 call DumpRegs
```
- Diagnostic Tools:** Diagnostics session: 0 seconds (14 ms selected). Events: 12ms. Process Memory: Summary, Events, Memory Usage, CPU Usage. Show Events (10 of 10).